基于搜索的分层回归测试数据集扩增方法 *

王曙燕，高　露，孙家泽

(西安邮电大学 计算机学院，西安 710121)

摘　要：针对在回归测试中原有的测试数据集往往难以满足新版本软件的测试需求问题，提出一种基于搜索的分层回归测试数据集扩增方法，主要包含覆盖目标方法集获取模块和测试数据生成模块。首先对新版本程序进行抽象分析，提取出方法调用图，利用方法调用轨迹和已有测试数据建立方法覆盖信息，获取目标方法集，并通过计算贝叶斯条件概率对目标方法集进行优先选择；利用 Hadamard 矩阵设计正交种群，同时结合已有测试数据集进行种群初始化，采用文化基因算法对目标集中方法生成测试数据。该方法针对四个基准程序与随机法和遗传算法以及基于粒子群算法测试数据生成方法相比，测试数据的生成效率平均提高了 95.2%、78.2%和 50.5%，测试数据检错能力平均提高了 47.9%、33.6%和 18.2%，实验结果表明，该方法更适合回归测试数据扩增。

关键词：方法调用图；回归测试数据扩增；正交种群；文化基因算法

中图分类号：TP311.5　　doi: 10.3969/j.issn.1001-3695.2018.05.0272

Search-based hierarchical regression test suite augmentation method

Wang Shuyan, Gao Lu, Sun Jiaze

(*School of Computer Science & Technology, Xi'an University of Posts & Telecommunications, Xi'an* 710121, *China*)

Abstract: It is difficult for the original test data to meet the requirements of the new version of software testing in regression testing, thus a search-based hierarchical regression test suite augmentation method was proposed to solve the problem. The method mainly includes obtain the target function set module and the test data generation module. Firstly, it **Abstract:** function call graph from the new program, and builds function coverage information use function traces and original test case set, afterwards, Bayesian conditional probability choose the target function set through calculating; Then using Hadamard matrix to design the orthogonal population, and initialization population with the combination of the orthogonal population and existing test data set. Finally, using the memetic algorithm to generate test data for target functions. Compared with the random algorithm based, the genetic algorithm based and particle swarm algorithm based test data generation methods on four benchmark programs, the generation efficiency of the proposed method was improved on average by approximately 95.2%, 78.2% and 50.5% respectively, the error detection ability of the test case was improved on average by approximately 47.9%, 33.6% and 18.2% respectively. The experimental results show that the proposed method is more suitable for regression test suite augmentation.

Key words: function call graph; regression test suite augmentation; orthogonal population; memetic algorithm

0　引言

软件测试是提高软件质量和软件可靠性的重要手段，同时也可提高被测软件的可信度[1,2]。回归测试是指测试人员通过重新测试新版本软件，以确定软件中没有引入新的错误或者其他潜在错误[3]。在软件生命周期中，由于频繁地对软件进行修改，使得原有测试数据集往往难以满足新版本软件的测试需求，从而难以保证覆盖到软件新增和发生改变的部分以及揭示软件中存在的错误或潜在问题。为了补充原有测试数据集的完备性，则需要对原有测试数据集进行扩增。据统计,回归测试开销一般占整个测试预算的 80%左右[4,5]，因此提高回归测试效率是非常有意义的。

最初，Menager 等人[6]提出了测试数据扩增的概念，当时提出的目的是为对应测试数据集的约减方法，因此并没有阐述应该如何进行回归测试数据集的扩增。近些年来，测试数据扩增方法主要分为两大类，即面向行为的回归测试数据扩增方法

收稿日期：2018-05-16；**修回日期：**2018-07-13　**基金项目：**陕西省工业攻关项目（2017GY-092）；西安邮电大学创新基金重点项目（CXJJ2017020）

作者简介：王曙燕（1964-），女，河南南阳人，教授，博士，主要研究方向为软件测试、数据挖掘、智能信息处理（wsylxj@126.com）；高露（1993-），女，陕西西安人，硕士研究生，主要研究方向为软件设计与测试、数据挖掘；孙家泽（1980-），男，河南南阳人，副教授，博士，主要研究方向为软件测试、数据挖掘、智能信息处理.

和面向覆盖的回归测试数据扩增方法，前者主要期望通过生成的测试数据集来描述软件新增部分和修改部分在执行上的变化；后者期望生成的测试数据集可以覆盖到软件新增部分和修改部分。

Santelices 等人[7]利用符号执行技术分析软件的控制流图和数据流图进行测试数据的扩增，提出回归测试扩增技术应满足状态需求和链需求；随后将 PIE 模型引入到回归测试数据集扩增方法中，进一步再利用符号执行和约束求解获得测试数据集；Xu 等人[8]提出一种导向测试数据集扩增技术，主要通过分析测试数据的再执行，来获取分支覆盖目标，即未被任何测试数据覆盖到的分支集合，再结合符号执行技术生成新的测试数据；Xu 等人[9]进一步遗传算法引入测试数据扩增技术中，并分析了遗传算法参数的设置对扩增结果的影响；Zhang 等人[10]提出了一种基于代码的回归测试方法，用于为单元测试生成测试数据，主要借助分析程序的树形结构以便找到发生改变的路径，将分支路径作为测试数据生成主要目标。Xu 等人[11]利用原有测试数据运行修改后的程序，以发生改变的语句作为目标语句，初始种群是根据目标语句的个数在已有测试数据集中进行选择，最后利用遗传算法生成覆盖目标语句的测试数据。巩敦卫等人[12]分析原有测试数据穿越的路径与目标路径的相似度，来选取部分测试数据集作为部分初始种群，利用遗传算法进化生成测试数据；吴川等人[13]着重于考虑已有测试数据集的利用，提出基于分析路径相关性来建立数据生成数学模型并用遗传算法进行模型求解。王曙燕等人[14]提出一种基于自适应粒子群算法的回归测试扩增方法，利用路径相似度识别新增路径并选取初始种群，然后建立测试数据集扩增数学模型，最后采用自适应粒子群进行求解。

在上述回归测试数据生成的方法中，都将测试数据的覆盖率作为测试数据生成的最终目标，而忽略了测试数据集的缺陷检测能力。测试数据作用是确保用尽量少的测试数据去发现程序去更多的缺陷。本文从测试数据的覆盖率和检错能力两方面考虑，在文献[14]工作的基础上，针对需要进行回归测试的两类场景生成测试数据，一类是软件系统新增若干方法体的回归测试，一类是程序缺陷修复后进行的回归测试，提出一种基于搜索的分层回归测试数据扩增方法，本文的分层是指依次从方法级别和语句级别提取覆盖目标，即先从方法级别提取目标方法集，包括新增或者发生改变的方法，再针对目标方法集进行静态分析提取覆盖路径，以期降低识别提取覆盖目标的效率。利用贝叶斯理论对覆盖目标方法集进行优先选择，针对目标路径集利用文化基因算法（memetic algorithm, MA）生成测试数据，以期提高测试数据的覆盖率和检错能力，从而降低回归测试的测试成本。本文方法框架，如图 1 所示。

1　获取覆盖目标方法集

首先得到　程序的方法调用图，将其用邻接矩阵存储，计算所有节点出边的平均执行概率，得到加权方法调用图，根据方法调用轨迹和已有测试数据建立方法覆盖信息，根据方法覆盖信息获取得到覆盖目标方法集，依据贝叶斯理论计算选择可能被引入错误方法，得到优先覆盖目标方法，即新增方法和修复缺陷后可能存在错误的方法。

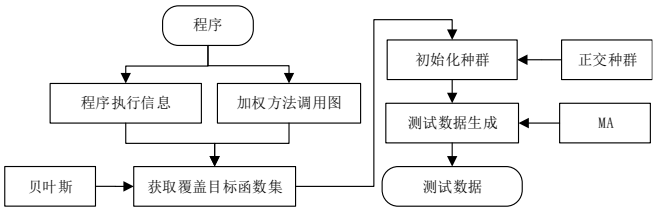

图 1　本文方法框架图

1.1　加权方法调用图

方法调用图为一个有向图，记作 G，有向边记作 e。其中，每条边 e 附加有一个权重 w。本文采用边的平均执行概率作为权重。假设方法调用图有 n 条边，每条边在执行路径中出现的次数分别是 $u_1,u_2,\cdots u_n$，则边 e_i 的权值 w_i 的公式为

$$w_i=\frac{u_i}{\sum_{j=0}^{n}u_j}$$

对于同一个定点的出边的权值和为 1。

$$\sum_{i=0}^{n}w_i=1$$

现在有方法调用图如图 2 所示。

有测试用例 $ts_1=(8,10,13)$，其方法覆盖路径为 main，f_1，f_2，f_4，f_6，f_2，f_4，f_6，f_2，f_3，f_6，f_2，f_7，f_8，对于 ts_1 在执行时，方法 f_1 只能调用方法 f_2，因此 $f_1\rightarrow f_2$ 权值为 1，对于方法 f_2，其可调用 f_3，f_4 或 f_7，调用边 $f_2\rightarrow f_4$，$f_2\rightarrow f_3$ 和 $f_2\rightarrow f_7$，分别出现的次数为 2 次，1 次和 1 次，因此 $f_2\rightarrow f_4$ 边的权值为 2/(2+1+1)=0.5，$f_2\rightarrow f_3$ 边的权值为 1/(2+1+1)=0.25，$f_2\rightarrow f_7$ 边的权值为 1-0.5-0.25=0.25，因此可以得到在 ts_1 执行时方法的加权方法调用图如图 3 所示。计算测试数据集在方法调用图上每个节点出边的权值，得到每个节点出边的最终权值为该出边在不同测试数据执行时的调用概率的平均值。

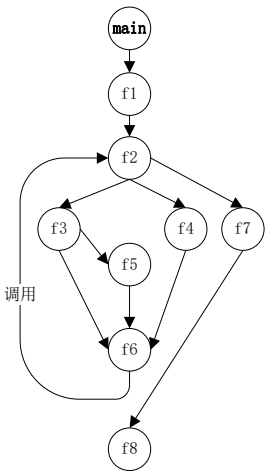

图 2　方法调用

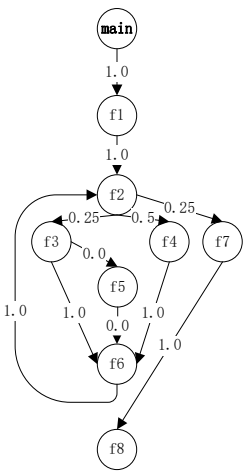

图 3　加权方法调用图

1.2 方法覆盖信息

假定一个软件系统由 n 个组件 C_i 组成，其中 $i \in \{1,\cdots,n\}$，测试执行集合 $T = T_F \cup T_P$，其中 T_F 表示测试运行结果错误的测试用例集合，T_P 表示测试执行结果正确的测试用例集合，则程序覆盖信息是一个 2 维矩阵。

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & a_{ij} & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}$$

其中：第 i $(1 \le i \le m)$ 行为方法 $f_1, f_2, \cdots, f_m$，第 j $(1 \le j \le n)$ 列为测试数据 $t_1, t_2, \cdots, t_n$，如果测试数据 t_j 覆盖了方法 f_i，则 $a_{ij} = 1$，否则 $a_{ij} = 0$；用向量 $\boldsymbol{e} = [e_1, \cdots, e_i, \cdots e_m]$ 表示测试数据的执行结果，其中元素 e_i 表示第 j 个测试数据的执行结果，如果成功，则 $e_j = 1$，否则 $e_j = 0$；则 (A, e) 是本文需要的方法覆盖信息。

1.3 目标方法集选择

本文将方法覆盖信息 (A, e) 作为输入，得到目标方法集 $C_k = \{c_i \in A \mid a_{ij} = 0 \cap (a_{ij} = 1 \wedge e_i = 0)\}$，即新增方法集和修改后可能包含错误的方法集，其中 $k \in [1, m]$，然后通过贝叶斯概率计算出目标方法包含错误的概率，现在假定方法集是相互独立的，本文利用贝叶斯定理计算方法集 c_k 的条件概率为

$$P(c_k \mid e) = w_k \times \frac{P(e \mid c_k) P(c_k)}{p(e)} \tag{1}$$

其中：w_k 是第 k 个节点出边的执行概率，$P(e)$ 是一个标准化常数，由所有 c_k 定义，无需要直接计算。由于每次执行都是独立的，故

$$P(e \mid c_k) = \prod_{i=1}^{N} P(e_i \mid c_k) \tag{2}$$

其中，$P(e_i \mid c_k)$ 的定义如下：

$$P(v_i \mid c_k) = \begin{cases} 1, & c_k \text{与} v_i \text{不相容} \\ 0, & c_k \text{与} v_i \text{相关且唯一} \\ \varepsilon, & \text{其他} \end{cases} \tag{3}$$

其中，ε 定义为

$$\varepsilon = \begin{cases} \prod_{j \in c_k \cap a_{ij} = 1} h_j, & e_i = 0 \\ 1 - \prod_{j \in c_k \cap a_{ij} = 1} h_j, & e_i = 1 \end{cases} \tag{4}$$

其中：$h_j \in [0,1]$ 表示方法 j 不包含错误的概率，在上述定义中 $P(c_k) = p^{|c_k|} \times (1-p)^{M - |c_k|}$，假定 $p = 0.01$，即源程序大部分方法都没有包含错误，其中 M 为方法的个数，$|c_k|$ 是目标方法集 c_k 包含方法的个数。将 ε 和 $P(e_i \mid c_k)$ 代入 $P(e \mid c_k)$，可以得到 $P(e \mid c_k)$ 关于 h_j 的表达式，利用最大似然估计法计算 $H = \max_H (P(e \mid c_k))$，其中 $H = \{h_j \in [0,1] \mid j \in c_k\}$，将得到的 $P(e \mid c_k), P(c_k)$ 代入到计算公式（1）中，将 c_k 按照 $P(c_k \mid e)$ 值递增的顺序进行排列，选择出条件概率值排序较高的方法，优先对其生成测试数据。

2 测试数据生成

测试数据生成框架分为预处理模块(获取方法集和获取目标路径)、测试数据生成模块（文化基因算法），如图 4 所示，其中预处理模块是测试数据生成模块的前提，该模块主要负责前期获取有序覆盖目标方法集，以及对目标方法静态分析选择目标路径并提取相关参数进行算法参数初始化，再利用分支函数构造适应度函数；测试数据生成是核心模块，该部分通过根据种群的适应度值、种群评价、协作竞争算子来引导种群向目标解进化，最终生成覆盖目标方法集中目标路径的测试数据集。

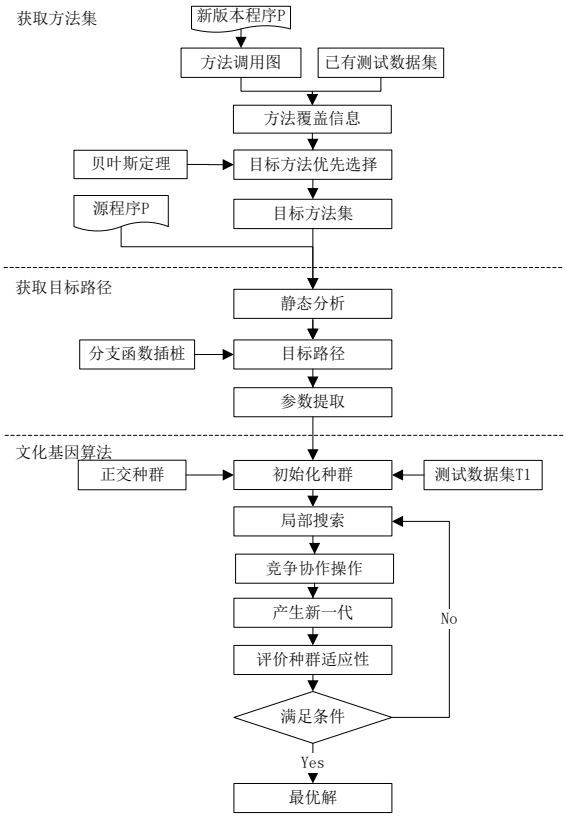

图 4 测试数据生成框架

2.1 初始化种群

在回归测试数据生成中利用已有测试数据集可以大幅度减少测试数据生成个数[14,15]，初始种群的多样性可保证启发式搜索算法在迭代进化过程中有较高的进化效率，有利于进化算法更加快速的走向全局最优[16]。但在算法进化迭代过程中，往往会由于后期种群的多样性减少导致出现早收敛问题，为了保证种群的多样性，从而提高回归测试数据生成效率，本文使用正交种群和部分原有测试数据集结合方式进行种群初始化。

正交设计法在 1949 年由田口玄一等人提出后一直广泛应用于寻优问题[17]。用正交设计法初始种群，可以使种群中个体更加均匀的分布在求解空间内，从而保证种群的多样性。根据 Zhang 等人将正交设计法用于遗传算法中保持种群的多样性提高解的最优性和算法收敛速度[18]，本文采用 Montgomery 等人[19]设计的正交设计法正交数组来进行文化基因算法的种群初始化，其中，正交数组表示为 $L_m(j^n)$，m 为正交数组的行数，表示测试次数或测试用例数，n 为正交数组的列数，表示被测对象的个数，j 为正交数组中的码数，表示被测对象的位级数，当位级数为 2 时，正交数组只有两种取值 0 或 1。目前构造正交数组的算法有行交换算法、坐标交换算法或者拟人拟物算法，本文通过 Hadamard 矩阵设计正交数组，因为这种方式可以快速高效地产生两位级数的正交数组[22]，一个 $n\times n$ 的矩阵表示为

$$\boldsymbol{H}\times\boldsymbol{H}^T=\boldsymbol{H}^T\times\boldsymbol{H}=n\boldsymbol{I}$$

其中:矩阵元素均为 1 或-1，$\boldsymbol{H}^T$ 为转置矩阵，$\boldsymbol{I}$ 为单位矩阵，则称 $\boldsymbol{H}$ 为一个 n 阶的 Hadamard 矩阵[19]-[24]。一个 4 阶的 Hadamard 矩阵为

$$\boldsymbol{H}_4=\begin{pmatrix}+1 & +1 & +1 & +1\\ +1 & -1 & +1 & -1\\ +1 & +1 & -1 & -1\\ +1 & -1 & -1 & +1\end{pmatrix}$$

如果 $\boldsymbol{H}$ 是一个阶数为 n 的 Hadamard 矩阵，那么一个 $2n$ 阶的 Hadamard 矩阵为

$$\boldsymbol{H}_{2n}=\begin{pmatrix}+\boldsymbol{H} & +\boldsymbol{H}\\ +\boldsymbol{H} & -\boldsymbol{H}\end{pmatrix}$$

首先，构造一个 k 阶的 Hadamard 矩阵 $\boldsymbol{H}_k$，其中 $k=\lceil(N+1)/4\rceil\times4$，$N\leftarrow max\{m,n\}$，$m$ 是初始种群数，n 是被测对象数；接着对 $\boldsymbol{H}_k$ 的行进行从小到大排序并且删除 $\boldsymbol{H}_k$ 中元素全为 0 的第一列；形成矩阵 $\boldsymbol{H}_{k\times k-1}$，并将 $\boldsymbol{H}_{k\times k-1}$ 转化成正交数组 $L_k(2^{k-1})$；最后采用正交数组 $L_k(j^{k-1})$ 设计正交种群，再使用正交种群和部分已有测试数据集进行种群初始化操作。

2.2 适应度函数的设计

个体的生存能力主要体现在个体的适应度，也是决定个体的生存竞争能力，因此设计合理的适应度函数具有至关重要的作用。本文采用分支距离法设计适应度函数，分支距离是指一个谓词为真（或为假）的条件满足程度[22]，本文的目标是覆盖目标路径，即目标路径中每个分支节点，通过在各个分支中插入分支函数，来表示目前测试数据与真分支的距离 $f(x)$，如果分支距离小于等于 0 表示该真分支被覆盖，则将分支距离 $f(x)$ 置为 0。假设该目标路径中含有 n 个分支，则适应度函数转化为标准化分支距离为

$$fitness(x)=1-1.001^{-f(x)} \tag{5}$$

$$f(x)=\frac{1}{n}\sum_{i=0}^{n-1}\frac{1}{f_i(x)+1}\times100\% \tag{6}$$

适应度值 $fitness(x)=0$ 的充要条件是 x 正好覆盖目标路径的各个分支。若 $fitness(x)$ 函数值越小，表示 x 越接近目标，因此覆盖目标路径的测试数据生成问题可以转化为适应度函数 $fitness(x)$ 的最小化问题。

2.3 文化基因算法

文化基因算法是一种基于种群的全局搜索和基于个体的局部启发式搜索的结合体[25]-[28]，其优势在于全局和局部结合的搜索策略[29]-[31]。将问题的解表示成“染色体”，通过模拟文化进化过程的竞争、协作操作结合局部搜索，实现个体适应度的提高，不断迭代，逐步寻找最优解或次优解。

本节以利用原程序 P 的测试数据集 TC 生成覆盖新版本程序 P' 方法集的测试数据为例，详细介绍本文方法的实现步骤：

输入：旧程序 P 的测试数据集 TC 和新版本程序 P'；

输出：覆盖 P' 方法集的测试数据集。

a)抽取程序方法调用图，结合原有测试数据集，运行新版本程序，建立方法覆盖信息，进而获取得到覆盖目标方法集；

b)依次选取目标方法集中的方法对其进行静态分析，得到覆盖目标路径集以及相关参数；

c)逐个选取目标路径，进行分支函数插桩，设定初始参数，设计正交种群，进行种群初始化 $Z\leftarrow p$；

d)根据适应度函数式(5)(6)计算种群 p 的适应度值 $fitness(p)$，评价种群；

e)判断是否满足终止条件，即生成的测试数据覆盖 P' 的目标路径，或达到最大进化代数，如果满足终止条件，转步骤 i)，否则转步骤 f)；

f)选择父代个体 p_1，p_2，由父代个体通过交叉算子和变异算子产生子代个体 o_1，o_2，计算父代和子代适应度值及

$$f_p=\min\left(fitness(p_1),fitness(p_2)\right)$$

和

$$f_o=\min\left(fitness(o_1),fitness(o_2)\right);$$

g) 如果 $f_p > f_o$，则 $Z \leftarrow o_1, o_2 \cup Z$，否则，按照 $\varepsilon \in Random(0,1) > \gamma = exp(\Delta o / T)$ 接受子代个体，其中 $\Delta o = f_p - f_o$，T 为局部策略中温度；

h)通过变异算子产生新一代种群，转步骤 d)；

i)算法结束，输出测试数据集。

3 实例分析

通过三角形判定程序来说明本文的方法。在原程序的测试数据集中选取测试数据为 $t_1 = (1,2,3)$ 和 $t_2 = (3,4,5)$ 对本文方法进行说明。首先通过工具 Doxygen 抽取新旧版本程序方法调用图，再通过本文 1.1 节的方式计算新版本程序方法调用图中每个节点出边的平均权重，可得到加权方法调用图，旧版本方法调用图和新版本加权方法调用图，如图 5 所示。

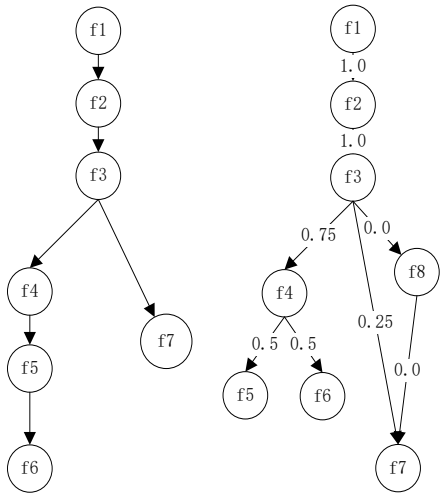

图 5 旧版本和新版本程序的方法调用图

其中，方法 $f1-f8$ 分别为：$main$ 方法、构造方法、获取三角形类型、判断三边是否符合三角形定义、获得三边长度、计算两边之差、输出方法、判断三角形类型。其中，在新版本程序中，f_6 正确程序为

$$f_6(a,b)\{\cdots if(a>b)\cdots\},$$

修复缺陷后包含错误的程序为

$$f_6(a,b)\{\cdots if(a\le b)\cdots\}。$$

分析测试数据 t_1, t_2 在新旧版本方法覆盖信息，如表 1 所示。

表 1 新旧版本方法覆盖信息

	P_{old}		P_{old}		P_{new}		P_{new}	
方法	t_1	e_1	t_2	e_2	t_1	e_1	t_2	e_2
$f1 \rightarrow main()$	1	1	1	1	1	1	1	1
$f2 \rightarrow Triangle()$	1	1	1	1	1	1	1	1
$f3 \rightarrow getType()$	1	1	1	1	1	1	1	1
$f4 \rightarrow isTriangle()$	1	1	1	1	1	0	1	0
$f5 \rightarrow getBorders()$	0	0	1	1	0	0	1	1
$f6 \rightarrow diffOfBorders()$	0	0	1	1	0	0	1	0
$f7 \rightarrow pType()$	1	1	1	1	1	1	1	1
$f8 \rightarrow judgeType()$	无	无	无	无	0	0	0	0

根据新旧版本方法覆盖信息，可得到覆盖目标方法集为 $c_k = \{< f_4, f_6 >< f_8 >\}$，通过最大似然估计得到 $h_4 = 0.6, h_6 = 0.8, h_8 = 0$，再通过公式 (1) 计算得到：$P(c_4 \mid e) = 0.042367$，$P(c_6 \mid e) = 0.372826$，$P(c_8 \mid e) = 0$。因此覆盖方法集为 $C_k = \{f_6, f_4, f_8\}$；将 C_k 和源码作为测试数据生成模块的输入，优先对 f_6 进行静态分析，获取到被测对象数 $n = 3$ 和理论路径集 $Path = \{p_1, p_2\}$，依次选择 p_1, p_2 为目标路径，插入分支函数；采用 2.1 节方法设计初始种群，进行种群初始化，评价种群适应性，若满足输出条件，则结束算法，否则继进行迭代，直到生成覆盖目标路径的测试数据或达到最大迭代次数为止。

4 实验与分析

4.1 实验设置

为验证本文方法的有效性，选取四个基准程序，分别是三角形判定分类程序 Triangle、西门子工业程序集中的 Schedule 和 Tcas 以及程 NextDay。其基本信息如表 2 所示。

表 2 被测程序信息

程序	行数	方法数	实参数	输入范围
Triangle	50	8	3	[0,100]
Schedule	412	18	可变	[0,2047]
Tcas	138	9	12	[0,10000]
NextDay	60	2	3	[0,10000]

实验程序均用 Java 语言编写，计算机配置为 Windows Intel(R) Core(TM) i5-2450M CPU @ 2.50GHz，64 位操作系统，程序在 Eclipse MARS.2 4.5.2，jdk1.7.0_80 环境下运行。与本文方法相关的策略及参数设置如表 3 所示。针对不同程序独立运行 20 次，当生成覆盖到目标路径的测试数据或达到最大迭代次数时，终止算法。

表 3 本文方法参数设置

算法策略(参数)	取值
种群编码	二进制编码
局部策略	Metropolis 准则
选择策略	轮盘赌选择
交叉策略	单点随机交叉
交叉概率	0.9
变异策略	单点变异
变异概率	0.3
种群大小	30
最大迭代次数	1000

表 4 基准算法的参数设置

方法	算法参数	取值
随机法	随机种子	与输入范围相关
遗传法	变异概率	0.9
	交叉概率	0.3

粒子群法	学习因子	$c_1=c_2=2$
	惯性权重	$w_{max}=1.8, w_{min}=0$
	种群编码	二进制编码
公用参数	种群大小	30
	最大迭代次数	1000

4.2 实验结果

为验证本文方法的有效性，选择基于遗传算法（GA）和基于粒子群算法（PSO）以及随机法与本文方法进行比较，在相同条件下分别运行每种方法，记录其生成测试数据时的迭代次数和耗时，实验结果如表 5 和 6 所示。

表 5 不同方法生成测试数据时迭代次数

程序	指标	进化代数			
		GA	PSO	随机	本文方法
Triangle	均值	9.0	6.5	50.3	3.3
	标准差	8.6	3.2	19.7	0.9
Schedule	均值	678.4	145.2	1000.0	73.8
	标准差	27.3	16.5	0.0	7.2
Tcas	均值	189.3	49.1	1000.0	26.9
	标准差	35.1	7.3	0.0	3.4
NextDay	均值	57.4	28.8	519.5	14.1
	标准差	31.2	5.4	120.8	2.1

表 6 不同方法生成测试数据时的耗时

程序	指标	运行时间/s			
		GA	PSO	随机	本文方法
Triangle	均值	0.228	0.172	0.892	0.072
	标准差	0.209	0.056	0.224	0.028
Schedule	均值	9.112	6.117	16.341	4.172
	标准差	0.948	0.423	0.513	0.394
Tcas	均值	6.513	3.235	11.021	0.972
	标准差	0.518	0.201	0.410	0.178
NextDay	均值	1.479	0.336	8.132	0.286
	标准差	0.413	0.072	2.571	0.051

以表 5 中空中防撞系统 Tcas 为例，相比遗传算法需要平均迭代 189.3 次、运行时间 6.513 s，粒子群算法平均迭代 49.1 次、运行时间 3.235 s，以及随机法未能在规定的最大迭代次数中完成测试要求，本文方法只需要迭代 26.9 次、运行时间 0.972 s 即可生成覆盖目标方法体中目标路径的测试数据。综合各种生成方法在不同基准程序的生成效率，本文方法与基于粒子群算法和基于遗传算法以及随机法的测试数据生成方法相比，测试数据生成效率平均分别提高了约 50.5% 和 78.2%以及 95.2%。由此可看出本文方法生成回归测试数据的效率。而且，通过多次运行结果求得的标准差可以看出，本文方法在稳定性方面也具有较好的优势。

同时从测试数据的覆盖率和缺陷检测方面评价不同回归测试数据生成方法的能力。以空中防撞系统 Tcas 为例，不同方法生成的测试数据路径覆盖率如图 6 所示。由图 6 可以看出，本文方法与其他三种方法相比在路径覆盖率上也有明显的优势。

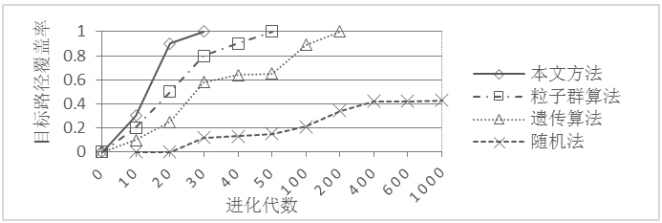

图 6 针对 Tcas 生成测试数据覆盖率对比图

在相同条件下，在实验源码中使用 MuJava 工具注入变异体，使用的变异算子如表 7 所示。

表 7 变异算子描述

变异算子	描述	变异算子	描述
AOI	算术运算符插入	LOI	逻辑运算符插入
ROR	关系运算符替代	SDL	语句删除
COR	条件运算符替代	VDL	变量删除
COI	条件运算符插入	ODL	运算符删除

用四种方式生成的测试数据分别运行变异后程序，其中变异程序所使用的变异算子以及测试数据个数如表 8 所示。

表 8 实验程序描述

程序	变异算子	原测试数据集	现测试数据集
Triangle	AOI,SDL,ODL,ROR,COR,LOI	60	100
Schedule	AOI,SDL,ROR,LOI,COI,ODL	300	380
Tcas	AOI,SDL,ODL,ROR,COR,LOI,COI,SDI	210	290
NextDay	AOI,SDL,ROR,LOI	120	180

当一个测试数据能够从结果上区分原程序和某变异体，则称该变异体被杀死，剩下未被测试数据杀死的变异体称为存活变异体，其中部分存活变异体可通过完善测试数据集来进一步杀死变异体，其中也存在一定的比例的变异体，在语法上和原程序有区别，但语义上和原程序完全等价，因此不能被任何测试数据杀死，这类变异体被称为等价变异体，本文采用变异测试评价测试数据集的检错能力，先利用原有测试数据集对变异程序进行执行，实验结果如表 9 所示，再利用扩充后的测试数据集来执行变异程序，实验结果如表 10 所示，通过表 9 和 10 的实验结果，可以明显看出扩充后的测试数据集可以提高被杀死变异的个数，从而降低了可能等价变异体个数。

表 9 实验结果描述(原有测试数据集)

程序	变异体数	杀死变异体个数	存活变异体个数
Triangle	325	125	200
Schedule	622	204	418
Tcas	482	167	315
NextDay	340	128	212

表 10 实验结果描述(现有测试数据集)

程序	变异体数	方法	杀死变异体个数	可能等价变异体

Triangle	325	随机法	195	130
		遗传法	260	65
		粒子群法	293	32
		本文方法	**325**	**0**
Schedule	622	随机法	249	373
		遗传法	311	311
		粒子群法	373	249
		本文方法	**498**	**124**
Tcas	482	随机法	222	260
		遗传法	261	221
		粒子群法	338	144
		本文方法	**434**	**48**
NextDay	340	随机法	170	170
		遗传法	204	136
		粒子群法	272	68
		本文方法	**323**	**17**

通过实验结果计算得出各个程序的变异分数，其中，变异分数=（杀死的变异体个数/注入的变异体个数）* 100，得出平均变异分数如表 11 所示，本文方法与随机法、遗传法和粒子群法相比，测试数据检错能力平均分别提高了约 47.9% 和 33.6% 以及 18.2%。

表 11　变异分数

程序	变异分数（平均）			
	随机法	遗传法	粒子群法	本文方法
Triangle	60	80	90	100
Schedule	40	50	60	80
Tcas	46	54	70	90
NextDay	50	60	80	95

5　结束语

在回归测试中扩充原有测试数据集的完备性是回归测试数据生成需要解决的关键问题，有效地分析提取出发生改变或者新增的部分是回归测试数据生成的前提，如何保证扩充后的测试数据集的高覆盖率和数据集的检错能力是提高回归测试效率的关键点，本文提出一种基于搜索的分层回归测试数据扩增方法，该方法通过对多个基准程序和西门子工业程序进行测试，并与随机法、利用遗传算法和粒子群算法的回归测试数据生成方法进行对比，实验表明，本文所提出的方法生成的测试数据在覆盖率和测试数据检错能力都有明显的提高。

在回归测试数据生成过程中，利用程序在方法粒度的方法覆盖信息分析出发生改变和新增的方法，再利用贝叶斯理论确认方法体中可能包含错误的概率，对包含错误可能性大的方法和新增方法分析生成覆盖目标路径集的测试数据集，在一定程度上提高了测试数据集的覆盖率和检错能力。在本文研究的基础上，下一步主要的工作包括两个方面，其一是重点扩大实验规模；其二是将本文方法用在大型复杂的实际软件中，利用软件演化信息高效准备的分析出覆盖目标，进而对覆盖目标进行测试数据生成，将是下一工作中要重点解决的两个问题。

参考文献：

[1] Qiu XiaoKang, Li Xuandong. A path-oriented tool supporting for testing [J]. Acta Electronica Sinica, 2004, 32 (S1): 235-237.

[2] 徐仁佐. 软件可靠性工程 [M]. 北京: 清华大学出版社, 2007. (Xu Renzuo. Software reliability engineering [M]. Beijing: Tsinghua University Press, 2007.)

[3] 张智轶, 陈振宇, 徐宝文, 等. 测试用例演化研究进展 [J]. 软件学报, 2013, 24 (4): 663-674. (Zhang Zhiyi, Chen Zhenyu, Xu Baowen, *et al.* Research progress on test case evolution [J]. Journal of Software, 2013, 24 (4): 663-674.)

[4] Beizer B, Software testing techniques [M]. NY: Van Nostrand Reinhold, 1990.

[5] Harrold M J. Reduce, reuse, recycle, recover: Techniques for improved regression testing [C]// Proc of IEEE International Conference on Software Maintenance. Picataway, NJ: IEEE Press, 2009.

[6] Harder M, Mellen J, Ernst M D. Improving test suites via operational abstraction [C]// Proc of International Conference on Software Engineering. Picataway, NJ: IEEE Press, 2003: 60-71.

[7] Santelices R, Chittimalli P K, Apiwattanapong T, *et al.* Test-suite augmentation for evolving software [C]// Proc of IEEE//ACM International Conference on Automated Software Engineering. Picataway, NJ: IEEE Press, 2008: 218-227.

[8] Xu Z, Rothermel G. Directed test suite augmentation [C]// Proc of Asia-Pacific Software Engineering Conference. Picataway, NJ: IEEE Press, 2011: 1110-1113.

[9] Xu Z, Cohen M B, Rothermel G. Factors affecting the use of genetic algorithms in test suite augmentation [C]// Proc of Conference on Genetic and Evolutionary Computation. New York: ACM Press, 2010: 1365-1372.

[10] Zhang Zhihao, Huang Jun, Zhang Bo, *et al.* Regression Test Generation Approach Based on Tree-Structured Analysis [C]// Proc of International Conference on Computational Science and ITS Applications. Picataway, NJ: IEEE Press, 2010: 244-249.

[11] Xu Z, Kim Y, Kim M, *et al.* Directed test suite augmentation: techniques and tradeoffs [C]// Proc of the 18th ACM SIGSOFT International Symposium on Foundations of Software Engineering. New York: ACM Press, 2010: 257-266.

[12] 巩敦卫, 任丽娜. 回归测试数据进化生成 [J]. 计算机学报, 2014, 37 (3): 489-499. (Gong Dunwei, Ren Lina. Evolutionary generation of regression test data [J]. Chinese Journal of Computers, 2014, 37 (3): 489-499.)

[13] 吴川, 巩敦卫. 基于路径相关性的回归测试数据进化生成 [J]. 计算机学报, 2015, 38 (11): 2247-2261. (Wu Chuan, Gong Dunwei. Evolutionary generation of test data for regression testing based on path correlation [J]. Chinese Journal of Computers, 2015, 38 (11): 2247-2261.)

[14] 王曙燕，温春琰，孙家泽. 基于自适应粒子群优化算法的测试数据扩增方法 [J]. 计算机应用，2016，36 (9): 2492-2496. (Wang Shuyan Wen Chunyan, Sun Jiaze. Test data augmentation method based on adaptive particle swarm optimization algorithm [J]. Journal of Computer Applications, 2016, 36 (9): 2492-2496.)

[15] 温春琰. 面向回归测试的测试数据扩增方法研究 [D]. 西安：西安邮电大学，2017. (Wen Chunyan. Research on Test Data Augmentation for Regression Testing [D]. Xi ’ an: Xi ’ an University of Posts & Telecommunications, 2017.)

[16] Maaranen H, Miettinen K, Penttinen A. On initial populations of a genetic algorithm for continuous optimization problems [J]. Journal of Global Optimization, 2007, 37 (3): 405.

[17] 陈理国，蔡之华. 改进的正交遗传算法及其在函数优化中的应用 [J]. 计算机工程与设计，2008，29 (6): 3413-3418. (Chen Ligu, Cai Zhihua. Application of improving orthogonal-based genetic algorithm in function optimization [J]. Computer Engineering & Design, 2008, 29 (6): 3413-3418.)

[18] Zhang Qingfu, Leung Yiuwing. An orthogonal genetic algorithm for multimedia multicast routing [J]. IEEE Trans on Evolutionary Computation, 1999, 3 (1): 53-62.

[19] Montgomery D C. Design and analysis of experiments [M]. NY: Wiley, 2007.

[20] Craigen R. Hadamard Matrices and Designs [M]// Combinatorial Designs. NY: Springer, 2004: 73-100.

[21] 王万江. Hadamard 矩阵的构造及其应用 [D]. 天津：天津工业大学，2007. (Wang Wgnjiang. The Construction of Hadamard Matrix and Its

[22] Application, Tianjin: Tianjin University of Technology, 2007.)

[23] Stinson D R. Combinatorial designs: constructions and analysis [M]. Berlin: Springer-Verlag, 2003.

[24] Seberry J. Orthogonal designs: hadamard matrices, quadratic forms and algebras [M]. Berlin: Springer-Verlag, 2017.

[25] Crnković D, Egan R, Švob A. Orbit matrices of Hadamard matrices and related codes [J]. Discrete Mathematics, 2018, 341 (5): 1199-1209.

[26] 姜淑娟，王令赛，薛猛，等. 基于模式组合的粒子群优化测试用例生成方法 [J]. 软件学报，2016，27 (4): 785-801. (Jiang Shujuan, Wang Lingsai, Xue Meng, *et al*. Test case generation based on combination of schema using particle swarm optimization [J]. Journal of Software, 2016, 27 (4): 785-801.)

[27] 刘漫丹. 文化基因算法研究进展 [J]. 自动化技术与应用，2007 (11): 14-18. (Liu Mandan. The development of the memetic algorithm [J]. Control Theory and Applications, 2007 (11): 14-18.)

[28] Fraser G, Arcuri A, Mcminn P. Test suite generation with memetic algorithms [C]// Proc of International Conference on Genetic and Evolutionary Computation. 2013: 1437-1444.

[29] Fraser G, Arcuri A, Mcminn P. A memetic algorithm for whole test suite generation [J]. Journal of Systems & Software, 2015, 103 (2): 311-327.

[30] Mundade A A, Pattewar T M. Test suite generation using Memetic algorithm on adaptive local search [C]// Proc of International Conference on Communications and Signal Processing. Picataway, NJ: IEEE Press, 2015: 0630-0633.

[31] Nejad F M, Akbari R, Dejam M M. Using memetic algorithms for test case prioritization in model based software testing [C]// Swarm Intelligence and Evolutionary Computation. Picataway, NJ: IEEE Press, 2016.

[32] Islam M M, Singh H K, Ray T, *et al*. An enhanced memetic algorithm for single-objective bilevel optimization problems [J]. Evolutionary Computation, 2017, 25 (4): 607-642.